AAS 22-104

VARIABLE RESOLUTION QUADTREE MAPPING FOR PLANETARY LANDING USING PLANAR ELEMENTS

Corey Marcus[*], Timothy Setterfield[†], and Renato Zanetti[‡]

We present a terrain mapping technique for autonomous spacecraft landing on planetary bodies. An estimate of the terrain is formed without any *a priori* knowledge using generic range and bearing measurements. The map is represented with the quadtree, a variable resolution discretization. This data structure allows high resolution representations in areas with complex terrain features and low resolution representation in areas with sparse measurements or simple terrain features. Each quadtree cell contains a planar estimate of its terrain, these planar estimates are initialized with a Maximum Likelihood estimate and refined with an Extended Kalman Filter. The planar estimation strategy is validated through Monte Carlo and the mapping technique is demonstrated with an planetary landing simulation based on real flight tests conducted as part of NASA's ALHAT program.

INTRODUCTION

On Mars it is colloquially known as the "seven minutes of terror;" entry, descent, and landing (EDL) is one of the most technically challenging phases of any robotic lander's mission. *A priori* knowledge of the terrain in which the spacecraft is landing may not be sufficient to accurately locate all potential landing hazards. Communication blackouts and lengthy light-time delays necessitate that the spacecraft autonomously identify landing hazards and landing sites during descent. Measurements obtained during EDL can be used to build or refine maps of the surface from which hazards or safe landing sites can be identified. These challenges are not unique to Mars and may be found in many planetary landing scenarios.

Most existing planetary landing mapping techniques are based off of fixed resolution Digital Elevation Maps (DEMs).[1,2] At low resolution, these maps fail to capture nuances and fine details of the terrain. At high resolution, they become inefficient representations of relatively flat and featureless portions of the terrain. Setterfield *et al*[3] address these limitations through a mapping structure based on the quadtree.[4] Quadtrees are a data structure which discretize two dimensional space with a tree of links and cells. The spatial domain of each cell can be divided into four child cells of equal area. This division can be continuously repeated, allowing various resolutions across the domain of the tree. Setterfield *et al* assign an elevation estimate to each cell of the tree. This structure allows the map to dynamically increase its resolution in areas with a high measurement density.

[*]PhD Student, Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, cmarcus@utexas.edu.

[†]Guidance and Control Engineer, Jet Propulsion Laboratory, California Institute of Technology

[‡]Assistant Professor, Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin.

Mapping is performed for a variety of reasons during descent. It is most often performed to locate previously unknown hazards while the spacecraft navigates to one of several predetermined landing sites as in NASA's Autonomous Landing and Hazard Avoidance Technology (ALHAT) and Safe and Precise Landing (SPLICE) programs.[5,6] The Mars2020 mission located the lander with respect to an *a priori* hazard map.[7] Future missions to environments obscured from orbital imagery such as the hazy surface of Titan or permanent shade of polar Lunar craters may have limited knowledge of the surface before landing. Significant research is ongoing to select landing sites based on maps generated or refined during descent.

Almost all landing site selection algorithms have one commonality; they depend on fitting planes to the local terrain.[8–13] These planes are commonly used to evaluate the local terrain's slope and roughness (presence of small scale hazards). The burden of planar estimation is often placed on the site selection algorithm. Many authors use DEMs as measurements for their planar estimation scheme. In a planetary mapping scenario, these DEMs are themselves estimates and any planar estimation scheme would be ingesting estimated elevation values which may have correlated estimation errors. This generally violates navigation filter best practices.[14]

In this paper, we propose a variable resolution quadtree map in which each cell contains a planar estimate of the local terrain. Resolution is controlled by the degree to which the local terrain fits the planar approximation. This division scheme enhances the positive attributes of the quadtree. Complex areas of the terrain are represented in detail with fine resolution, and computational complexity is greatly reduced for simple or sparse segments of the terrain.

Modeling each cell as a planar approximation of the terrain produces a map which is of significant value to landing site selectors. Landing site selection complexity is greatly reduced by providing a map which already contains planar estimates and associated uncertainties. Furthermore, the theoretical foundation for planar estimates becomes much firmer by eliminating the need for sequential estimators.

Each cell's plane contains an estimate of a surface normal vector. Estimating such vectors results in many of the same problems found in attitude estimation. Namely, the parameterizations used to accommodate the length constraint contain singularities or redundant parameters. We estimate the unit-normal through the manifold encapsulation technique developed by Hertzberg[15] which avoids singularities and redundant parameters. New measurements are used to update planar elements with an Extended Kalman Filter (EKF).

RELATED WORK

The most similar work to this one is that of Setterfield *et al*.[3] However, they solve a complete Simultaneous Localization and Mapping (SLAM) problem using sparse optimization methods. They consider uncertainty in both the spacecraft state and map while we consider uncertainty only in the map and employ an EKF. While their work approximates the terrain in each cell with a constant elevation estimate, we approximate the terrain with a plane. In addition, Setterfield *et al* choose to divide their map based on measurement density while we opt for division based on the quality of the planar fit.

Several other authors in the SLAM domain have documented works which utilize planar elements in their maps. Viejo and Cazorla[16] extract planar elements from point clouds, they then use Iterative Closest Point[17] to align the planes and produce a SLAM solution. The planes are not stored in an organized structure, nor are they updated with a Bayesian filter. Several authors[18,19] present SLAM systems containing planar elements updated via EKF. However, the normal vectors of their planes are parameterized with angles in a spherical

coordinate system while we use a parameterization developed by Hertzberg.[15] Like Viejo and Cazorla, they do not organize planes into a structured data representation. Another recent plane-based SLAM system was demonstrated by Grant.[20] This system focuses on using the large volume of data produced by a LIDAR system. The system can also handle individual points when planes are not a good fit. Instead of solving the SLAM problem with an EKF, they formulate the problem as a factor-graph optimization and solve with iSAM2.[21]

Our manifold representation[15] has also seen use in another mapping application. Gallant and Marshall[22] used the same representation to map the orientation of local planar surfaces and provide sensor heading estimates. They did not attempt to estimate the positions of the surface or organize the surfaces into a structured representation.

PRELIMINARIES

Parameterization of Unit Vectors

In three dimensions, unit vectors lie on the unit sphere, S^2. S^2 exists in $\mathbb{R}^3$ but has only two degrees of freedom. In this work we parameterize S^2 as proposed by Hertzberg et al.[15] and further developed by Gallant[23] and Hewitt.[24] A unit vector $\boldsymbol{n} \in S^2$ is separated into a vector component, $\boldsymbol{\kappa} \in \mathbb{R}^2$, lying on the x-y plane and scalar component, $\lambda = \cos(\phi)$, in the z direction. Here, ϕ, is the angle between $\boldsymbol{n}$ and the z axis.

An identiy unit vector, $\boldsymbol{o} = [0,\, 0,\, 1]^T$ may be transformed to the unit vector $\boldsymbol{n}$ using a rotation matrix $\boldsymbol{T}^+(\boldsymbol{n})$.

$$\boldsymbol{n} = \boldsymbol{T}^+(\boldsymbol{n})\boldsymbol{o} = \begin{bmatrix} \boldsymbol{I}_{2\times 2} - \frac{\boldsymbol{\kappa}\boldsymbol{\kappa}^T}{1+\lambda} & \boldsymbol{\kappa} \\ -\boldsymbol{\kappa}^T & \lambda \end{bmatrix} \boldsymbol{o} \tag{1}$$

The inverse operation maps $\boldsymbol{n}$ to $\boldsymbol{o}$ and uses rotation matrix $\boldsymbol{T}^-(\boldsymbol{n}) = \left(\boldsymbol{T}^+(\boldsymbol{n})\right)^T$.

To eliminate redundant degrees of freedom, $\boldsymbol{\kappa}$ is normalized, $\hat{\boldsymbol{\kappa}} = \boldsymbol{\kappa}/||\boldsymbol{\kappa}||$. Next, a two dimensional vector, $\boldsymbol{\phi} = \phi\hat{\boldsymbol{\kappa}} \in s^2$, is defined. The new space, s^2, allows expression of unit vectors without redundancy. Additional operators, $\boxplus : S^2 \times s^2 \to S^2$ and $\boxminus : S^2 \times S^2 \to s^2$ mimic addition and subtraction of unit vectors.[23] This framework allows for small vectors, $\delta\boldsymbol{\phi}$, to perturb unit vectors, $\tilde{\boldsymbol{n}} = \boldsymbol{n} \boxplus \delta\boldsymbol{\phi}$. Alternatively, a metric representing the error between two unit vectors can be found as, $\delta\boldsymbol{\phi} = \tilde{\boldsymbol{n}} \boxminus \boldsymbol{n}$.

Terrain Measurements

In this work we frequently consider measurements of a terrain. In practice, such measurements often take the form of a range, $\rho \in \mathbb{R}$, and bearing, $\boldsymbol{b} \in S^2$. We assume that each of these measurements' true values, $\bar{\rho}$ and $\bar{\boldsymbol{b}}$, have been corrupted by noise in the following manner

$$\rho = \bar{\rho} + \nu_\rho \qquad \nu_\rho \sim \mathcal{N}\left(0,\, \Sigma_\rho\right) \tag{2}$$

$$\boldsymbol{b} = \bar{\boldsymbol{b}} \boxplus \boldsymbol{\nu}_{\boldsymbol{b}} \qquad \boldsymbol{\nu}_{\boldsymbol{b}} \sim \mathcal{N}\left(\boldsymbol{0}_{2\times 1},\, \boldsymbol{\Sigma}_{\boldsymbol{b}}\right) \tag{3}$$

It is useful to create a psuedo-measurement of some point, $\boldsymbol{m} \in \mathbb{R}^3$, which exists on the surface of the terrain. This is done as a function of ρ, $\boldsymbol{b}$, and the sensor location, $\boldsymbol{m}^o$.

$$\boldsymbol{m} = \boldsymbol{m}^o + \rho\boldsymbol{b} \tag{4}$$

By linearizing and neglecting higher order effects, uncertainty in ρ and $\boldsymbol{b}$ can be used to approximate the uncertainty of the psuedo-measurement as $\boldsymbol{R} = \text{cov}(\boldsymbol{m})$.

$$\boldsymbol{R} = \boldsymbol{T}(\boldsymbol{b})^{+} \begin{bmatrix} \boldsymbol{\Sigma_b} & 0 \\ 0 & \Sigma_\rho \end{bmatrix} \boldsymbol{T}(\boldsymbol{b})^{-} \tag{5}$$

We model the psuedo-measurement as

$$\boldsymbol{m} = \boldsymbol{m}^o + \rho \boldsymbol{b} + \boldsymbol{\nu_m} \tag{6}$$

$$\boldsymbol{\nu_m} \sim \mathcal{N}(\boldsymbol{0}_{3\times 1},\, \boldsymbol{R}). \tag{7}$$

METHODOLOGY

Maximum Likelihood Plane Estimation

Consider a plane, $\mathcal{P} \subset \mathbb{R}^3$, defined in Hessian Normal Form by a unit normal vector, $\boldsymbol{n}$, and orthogonal distance from the origin, d. All points, $\boldsymbol{p} \in \mathbb{R}$, lying on the plane satisfy the relationship $\boldsymbol{n}^T\boldsymbol{p} - d = 0$. The orthogonal distance, ϵ, between an arbitrary point, $\boldsymbol{m} \in \mathbb{R}^3$, and the plane is given by

$$\epsilon = \boldsymbol{n}^T\boldsymbol{m} - d. \tag{8}$$

Consider a set $\mathcal{I} = \{0, 1, \ldots, N_r - 1\}$ of independent point measurements, $\boldsymbol{m}_i$, of the planar surface. Each $\boldsymbol{m}_i$ has been corrupted by Gaussian noise, $\boldsymbol{\nu} \sim \mathcal{N}(\boldsymbol{0}_{3\times 1},\, \boldsymbol{R}_i)$. The Maximum Liklihood (ML) plane estimate, $\hat{\mathcal{P}}_{\text{ML}}$, can be found through a minimization of the negative log-likelihood.

$$\hat{\mathcal{P}}_{\text{ML}} = \arg\min_{\boldsymbol{n},d} \text{-}\log\left(p(\boldsymbol{m}_\mathcal{I}|\boldsymbol{n}, d)\right) \tag{9}$$

Independence of each measurement allows the following simplification.

$$\text{-}\log\left(p(\boldsymbol{m}_\mathcal{I}|\boldsymbol{n}, d)\right) = \text{-}\log\left(\prod_{i\in\mathcal{I}} p(\boldsymbol{m}_i|\boldsymbol{n}, d)\right) \tag{10}$$

Because the distribution of each measurement is Gaussian, the negative log-likelihood can be further simplified as

$$\text{-}\log\left(p(\boldsymbol{m}_\mathcal{I}|\boldsymbol{n}, d)\right) \propto \sum_{i\in\mathcal{I}} \frac{1}{2}\epsilon_i^2 \boldsymbol{n}^T\boldsymbol{R}_i^{-1}\boldsymbol{n}. \tag{11}$$

Equation (11) cannot be optimized in closed form outside the special case when $\boldsymbol{R} \propto \boldsymbol{I}_{3\times 3}$. Instead, incremental optimization must be used to find $\hat{\mathcal{P}}_{\text{ML}}$. To this end, we define $\boldsymbol{n}^j$ and d^j as the estimates at the j^{th} iteration of the optimization. If we allow $\boldsymbol{n}^j$ and d^j to be peturbed by $\delta\boldsymbol{\phi}$ and δd we find the following expression for error.

$$\epsilon_i = \left(n^j \boxplus \delta\boldsymbol{\phi}\right)^T \boldsymbol{m}_i - (d^j + \delta d) \tag{12}$$

The peturbation of the normal vector can be shown to be linearized as follows

$$\boldsymbol{n}^j \boxplus \delta\boldsymbol{\phi} \approx \boldsymbol{T}^+(\boldsymbol{n}^j) \begin{bmatrix} \delta\boldsymbol{\phi} \\ 1 \end{bmatrix} \tag{13}$$

where $\boldsymbol{T}^+$ is a rotation matrix as defined previously.

Substituting the Equation (13) into Equation (11) and employing linearization about $\boldsymbol{n}^j$ and d^j results in the following expression for ϵ_i.

$$\epsilon_i \approx A_i - \boldsymbol{D}_i \begin{bmatrix} \delta\boldsymbol{\phi} \\ d \end{bmatrix} \tag{14}$$

where

$$A_i = \boldsymbol{m}_i^T \boldsymbol{n}^j - d^j \tag{15}$$

$$\boldsymbol{D}_i = \begin{bmatrix} -\boldsymbol{m}_i^T \boldsymbol{T}^+(\boldsymbol{n}^j) \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{0}_{1\times 2} \end{bmatrix} & 1 \end{bmatrix} \tag{16}$$

Defining $\boldsymbol{\Delta} = \begin{bmatrix} \delta\boldsymbol{\phi}^T, & d \end{bmatrix}^T$, the cost function in Equation (11) can be simplified into a quadratic.

$$\sum_{i\in\mathcal{I}} = \frac{1}{2}\epsilon_i^2 \boldsymbol{n}^T \boldsymbol{R}_i^{-1} \boldsymbol{n} \tag{17}$$

$$= \frac{1}{2}a - \boldsymbol{\Delta}^T\boldsymbol{\beta} + \frac{1}{2}\boldsymbol{\Delta}^T\boldsymbol{H}\boldsymbol{\Delta} \tag{18}$$

where

$$a = \sum_{i\in\mathcal{I}} A_i^2 \Sigma_{\epsilon_i}^{-1} \tag{19}$$

$$\boldsymbol{\beta} = \sum_{i\in\mathcal{I}} \boldsymbol{D}_i^T \Sigma_{\epsilon_i}^{-1} A_i \tag{20}$$

$$\boldsymbol{H} = \sum_{i\in\mathcal{I}} \boldsymbol{D}_i^T \Sigma_{\epsilon_i}^{-1} \tag{21}$$

$$\Sigma_{\epsilon_i} = (\boldsymbol{n}^j)^T \boldsymbol{R}\boldsymbol{n}^j \tag{22}$$

Setting the derivative of the cost function to zero, the optimal increment, $\boldsymbol{\Delta}^*$, can be found by solving $\boldsymbol{H}\boldsymbol{\Delta}^* = \boldsymbol{\beta}$. The optimal values are then used to update the linearization points as follows:

$$\boldsymbol{n}^{j+1} = \boldsymbol{n}^j \boxplus \delta\boldsymbol{\phi}^*, \quad d^{j+1} = d^j + \delta d^* \tag{23}$$

This update process is repeated until convergence is achieved. At convergence, we can quantify uncertainty in $\hat{\mathcal{P}}_{\text{ML}}$ by considering the covariance of $\boldsymbol{\Delta}$. It can be shown that $\text{cov}(\boldsymbol{\Delta}) = \boldsymbol{H}^{-1} = \hat{\boldsymbol{P}}_{\text{ML}}$.

Extended Kalman Filter Plane Estimation

Post ML initialization, subsequent measurements can be used to refine the estimate. We use an Extended Kalman Filter (EKF) to produce a estimate of the error in $\hat{\mathcal{P}}$. This estimate will be used to update $\hat{\mathcal{P}}$ and account for the estimate error. Estimate error is represented by $\boldsymbol{\Delta} = \left[\delta\boldsymbol{\phi}^T,\, \delta d\right]^T$. It is assumed that $\boldsymbol{\Delta} \sim \mathcal{N}\left(\mathbf{0}_{3\times 1},\, \hat{\boldsymbol{P}}\right)$. The estimate is refined via a sensor at position m_0 producing range, ρ, and bearing, $\boldsymbol{b}$ measurements. These measurements are converted into a psuedo-measurement of a point on the plane, $\boldsymbol{m} \in \mathbb{R}^3$. This process can be repeated to form a recursive estimate of the plane, using the i^{th} psuedo-measurement, $\boldsymbol{m}_i$, we will update $\hat{\mathcal{P}}_i$ to produce $\hat{\mathcal{P}}_{i+1}$.

$\hat{\mathcal{P}}_i$ is expressed as an estimate of the unit normal vector, $\hat{\boldsymbol{n}}_i \in S^2$, and perpendicular distance from the origin, $d_i \in \mathbb{R}$.

$$\hat{\mathcal{P}}_i = \begin{bmatrix} \hat{\boldsymbol{n}}_i \\ d_i \end{bmatrix} \tag{24}$$

We define the state update, $\mathcal{U}$, as

$$\hat{\mathcal{P}}_{i+1} = \mathcal{U}\left(\hat{P}_i,\, \hat{\boldsymbol{x}}_{\boldsymbol{i}}\right) = \begin{bmatrix} \hat{\boldsymbol{n}}_i \boxplus \delta\phi_i \\ d_i + \delta d_i \end{bmatrix}. \tag{25}$$

We estimate $\boldsymbol{\Delta}_i$ using an EKF. The EKF produces $\boldsymbol{\Delta}_i$ as a function of its prior, $\bar{\boldsymbol{\Delta}}_i$. Because error is used to update $\hat{\mathcal{P}}_i$ in each time step we assume that $\bar{\boldsymbol{\Delta}}_i$ is normally distributed as $\bar{\boldsymbol{\Delta}}_i \sim \mathcal{N}\left(\mathbf{0}_{3\times 1},\, \hat{\boldsymbol{P}}_i\right)$.

Our psuedo-measurement, $\boldsymbol{m}_i \in \mathbb{R}^3$, is the location of a point on the surface of $\mathcal{P}$ at a given range, $\rho \in \mathbb{R}$, and bearing, $\boldsymbol{b} \in S^2$, as recorded by a sensor at $\boldsymbol{m}^o$. As discussed in a previous section, ρ, $\boldsymbol{b}$, and $\boldsymbol{m}^o$ all have an associated uncertainty which leads to uncertainty in the true value of $\boldsymbol{m}_i$. We model the uncertainty in $\boldsymbol{m}_i$ as zero-mean, additive Gaussian noise, $\boldsymbol{\nu}_i \sim \mathcal{N}\left(\mathbf{0}_{3\times 1},\, \boldsymbol{R}\right)$.

$$\boldsymbol{m}_i = \boldsymbol{m}^o + \rho\boldsymbol{b} + \boldsymbol{\nu}_i \tag{26}$$

The psuedo-measurement can be predicted as a function of $\hat{\mathcal{P}}_i$ and $\boldsymbol{\Delta}_i$ according to

$$\bar{\boldsymbol{m}}_i = h\left(\hat{\mathcal{P}}_i,\, \boldsymbol{\Delta}_i\right) = \boldsymbol{m}^o + \frac{\hat{d}_i + \delta d_i - \left(\hat{\boldsymbol{n}}_i \boxplus \delta\boldsymbol{\phi}_i,\, \boldsymbol{m}^o\right)}{\left(\hat{\boldsymbol{n}}_{\boldsymbol{i}} \boxplus \delta\boldsymbol{\phi}_i,\, \boldsymbol{b}\right)}\boldsymbol{b} \tag{27}$$

The EKF can now produce $\boldsymbol{\Delta}_i$ as

$$\boldsymbol{\Delta}_i = \hat{\boldsymbol{P}}_i\boldsymbol{H}^T\left(\boldsymbol{H}\hat{\boldsymbol{P}}_i\boldsymbol{H}^T + \boldsymbol{R}\right)^{-1}\left(\boldsymbol{m}_i - \bar{\boldsymbol{m}}_{\boldsymbol{i}}\right) \tag{28}$$

Where $\boldsymbol{H} = \frac{\partial h}{\partial \boldsymbol{\Delta}_i}|_{\boldsymbol{\Delta}_i = 0,\, \hat{\mathcal{P}}_i}$.

The updated error covariance is given in Joseph form by

$$\hat{\boldsymbol{P}}_{i+1} = \hat{\boldsymbol{P}}_i - \hat{\boldsymbol{P}}_i \boldsymbol{H}^T \left(\boldsymbol{H} \hat{\boldsymbol{P}}_i \boldsymbol{H}^T + \boldsymbol{R} \right)^{-1} \boldsymbol{H} \hat{\boldsymbol{P}}_i \tag{29}$$

Performance Evaluation Through Monte Carlo

To demonstrate the efficacy of this estimation technique, a 1000 run Monte Carlo simulation was performed. For each run a truth plane was created by perturbing a nominal plane $\mathcal{P} = \left[\boldsymbol{n}^T,\, d\right]^T$ according to

$$\boldsymbol{n}_i = \boldsymbol{n} \boxplus \delta\boldsymbol{\phi}_i \tag{30}$$

$$d_i = d + \delta d_i. \tag{31}$$

At each time-step, a sensor at random location $\boldsymbol{m}_i^o$ takes a measurement of $\mathcal{P}_i$. The measurement, $\boldsymbol{m}_i$, is along vector along vector $\boldsymbol{b}_i$.

$$\boldsymbol{\theta}_i = [0,\, 0,\, -1]^T + \delta\boldsymbol{b}_i \tag{32}$$

$$\boldsymbol{b}_i = \frac{\boldsymbol{\theta}_i}{||\boldsymbol{\theta}_i||} \tag{33}$$

Finally, the measurement is corrupted with random noise, $\boldsymbol{\nu}_i$.

$$\boldsymbol{m}_i = \boldsymbol{m}_i^o + \frac{d_i - (\boldsymbol{n}_i,\, \boldsymbol{m}_i^o)}{(\boldsymbol{n}_i,\, \boldsymbol{b}_i)} \boldsymbol{b}_i + \boldsymbol{\nu}_i. \tag{34}$$

The first 10 measurements are used to form an ML estimate of $\mathcal{P}_i$. 100 subsequent measurements, sensor locations, and apparent bearing vectors are used to update $\hat{\mathcal{P}}_i$ using an EKF. At each time-step, error is calculated as

$$\tilde{\boldsymbol{n}}_i = \boldsymbol{n}_i \boxminus \hat{\boldsymbol{n}}_{\boldsymbol{i}} \tag{35}$$

$$\tilde{d}_i = d_i - \hat{d}_i. \tag{36}$$

These errors are stored across all Monte Carlo runs to find the sample error mean and 3σ dispersion at each time step. These values are plotted alongside the predicted 3σ dispersion in Figure 1. This Figure shows the planar estimation technique is unbiased and consistent. Across all states, updates, and Monte Carlo runs, 1.5% of filter error values exceed the filter's predicted 3σ performance. This is lower than the expected 2.5% and implies that our filter actually underestimates its own performance. Specific values for the various distributions used in the Monte Carlo process are shown in Table 1.

Variable Resolution Planar Mapping

We use a QuadTree based map, $\mathcal{M}$, to model a surface, $\mathcal{S}$. The QuadTree is a variable resolution discretization of $\mathcal{S}$ in x-y space as shown in Figure 2. Each cell of the map contains a planar estimate of its portion of the terrain, $\mathcal{M} = \left\{\hat{\mathcal{P}}^1,\, \hat{\mathcal{P}}^2, \ldots, \hat{\mathcal{P}}^N\right\}$. Range and bearing measurements, ρ_i and $\boldsymbol{b}_i$, respectively, are collected from a sensor at $\boldsymbol{m}_i^o$ with each time-step. These measurements are used to create a psuedo-measurement, $\boldsymbol{m}_i$, of a point which lies on $\mathcal{S}$.

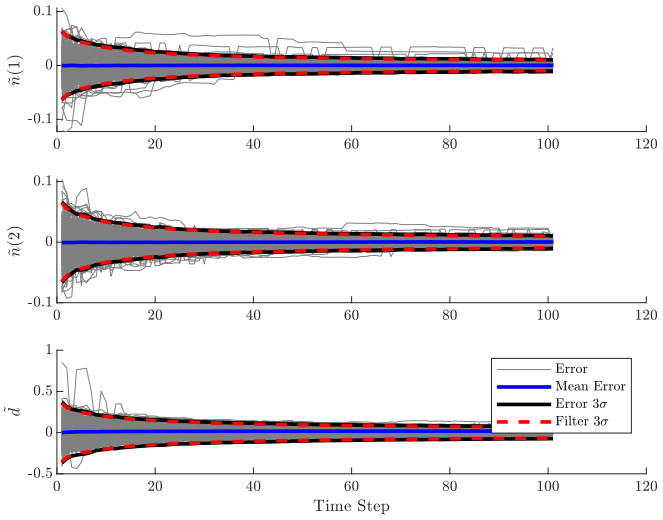


Figure 1: Aggregated Monte Carlo Results.

Table 1: Parameters for Monte Carlo Simulation

Variable	Value/Distribution
$\mathcal{P}$	$[0.099,\ 0.099,\ 0.990,\ 2.0]^T$
$\delta\boldsymbol{\phi}_i$	$\mathcal{N}\left(\mathbf{0}_{2\times1},\ 0.1\boldsymbol{I}_{2\times2}\right)$
δd	$\mathcal{N}\left(0,\ 0.1\right)$
$\delta\boldsymbol{b}_i$	$\mathcal{N}\left(\mathbf{0}_{3\times1},\ 0.2\boldsymbol{I}_{3\times3}\right)$
$\boldsymbol{m}_i^o$	$\mathcal{N}\left([0,\ 0,\ 10]^T,\ 10\boldsymbol{I}_{3\times3}\right)$
$\boldsymbol{\nu}_i$	$\mathcal{N}\left(\mathbf{0}_{3\times1},\ 0.1\boldsymbol{I}_{3\times3}\right)$

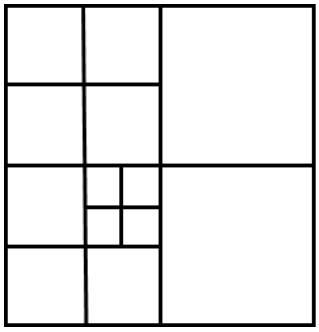


Figure 2: An example QuadTree showing the variable resolution discretization.

Each $\boldsymbol{m}_i$ is associated with an element, $\mathcal{P}^j \in \mathcal{M}$, according to its x and y coordinates. $\hat{\mathcal{P}}^j$ is initialized with an ML estimate once it has accumulated a sufficient number of psuedo-measurements. After initialization, $\hat{\mathcal{P}}^j$ is refined with any additional measurements using an EKF.

Cell division is based on psuedo-measurement residuals, $\boldsymbol{r}_i = \boldsymbol{m}_i - \bar{\boldsymbol{m}}_i$, and innovation covariance, $\boldsymbol{H}\hat{\boldsymbol{P}}_i\boldsymbol{H}^T + \boldsymbol{R}$, in the EKF update phase. For converged, linear, and Gaussian systems, it can be shown that $\boldsymbol{r}_i \sim \mathcal{N}\left(\boldsymbol{0}_{3\times 1},\, \boldsymbol{H}\hat{\boldsymbol{P}}_i\boldsymbol{H}^T + \boldsymbol{R}\right)$. We find the Mahalanobis distance for each residual.

$$\zeta_i = \sqrt{\boldsymbol{r}_i^T \left(\boldsymbol{H}\hat{\boldsymbol{P}}_i\boldsymbol{H}^T + \boldsymbol{R}\right)^{-1} \boldsymbol{r}_i^T} \tag{37}$$

We define the metric ψ^j as the average Mahalanobis distance of the N_m residuals used to update $\hat{P}^j$.

$$\psi^j = \frac{1}{N_m}\sum_{i=1}^{N_m} \zeta_i. \tag{38}$$

For a converged filter observing a perfectly planar $\mathcal{S}$ with a linear measurement model and Gaussian noise $E\left[\psi^j\right] = 1$. We assume that the estimate of $\hat{\mathcal{P}}^j$ has converged, our measurement models are sufficiently approximated through linearization, and that the measurement noise is roughly Gaussian. If the segment of $\mathcal{S}$ estimated by $\hat{\mathcal{P}}^j$ is poorly modeled by a plane then ψ^j will tend to be larger than one. We divide $\hat{\mathcal{P}}^j$ into four children whenever ψ^j is sufficiently large and the cell size is above a predefined minimum resolution.

SIMULATION

We simulate a spacecraft approaching a lunar hazard field which it would like to map. As it approaches, a flash LIDAR provides the mapping system with measurements of the terrain. The hazard field, spacecraft trajectory, and LIDAR performance are inspired by the flight tests conducted by Trawny *et al* as part of the ALHAT program.[2] These flight tests involve a rocket-powered vehicle approaching a man-made hazard field containing a mixture of rocks and craters which were representative of the Moon's surface. The vehicle scanned the field with a gimbal-mounted flash LIDAR and safe landing sites were identified.

Our simulated hazard field measures 60 by 60 meters and is diagrammed in Figure 3. It contains a mixture of rocks and craters which have the approximate size and locations of the rocks and craters in the field used by Trawny *et al*. Like the ALHAT test, our simulated vehicle takes 120 flash LIDAR images while approaching the field on a downward slope at a range of 490 to 400 meters. We run simulations at two approach angles, 30 and 60 degrees. The former is reminiscent of the trajectory flown by Trawny *et al* while the latter provides better observations of the terrain and enables better mapping performance.

Our simulated LIDAR has performance characteristics modeled after the one used in the ALHAT tests. Each simulated flash LIDAR image has a 1-degree field of view (FOV) and contains an array of 128x128 range measurements and bearing angles which are converted to psuedo-measurements as described in a previous section. We simulate the gimbaled motion of the real LIDAR so that the entire hazard field may be sampled. The vehicle's trajectories, hazard field, and LIDAR pointing targets are illustrated in Figure 4. The pseudo-measurements are corrupted with independent additive Gaussian noise with covariance $\boldsymbol{R}$. To approximate the real flash LIDAR's range precision of 8cm (1-σ), we choose

$$\boldsymbol{R} = 0.08^2\boldsymbol{I} \tag{39}$$

where $\boldsymbol{I}$ is the identity matrix. This results in a spherical uncertainty which is a computationally simple approximation of the true corruption of the psuedo-measurements in flight tests.

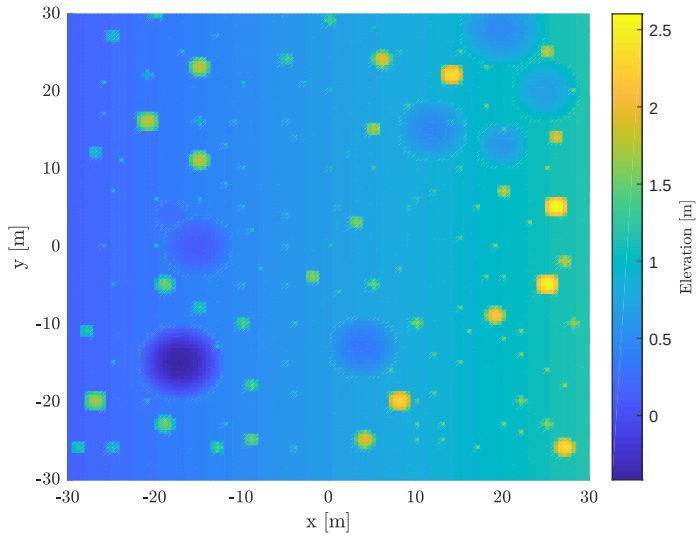


Figure 3: The simulated hazard field which contains a variety of rocks and craters.

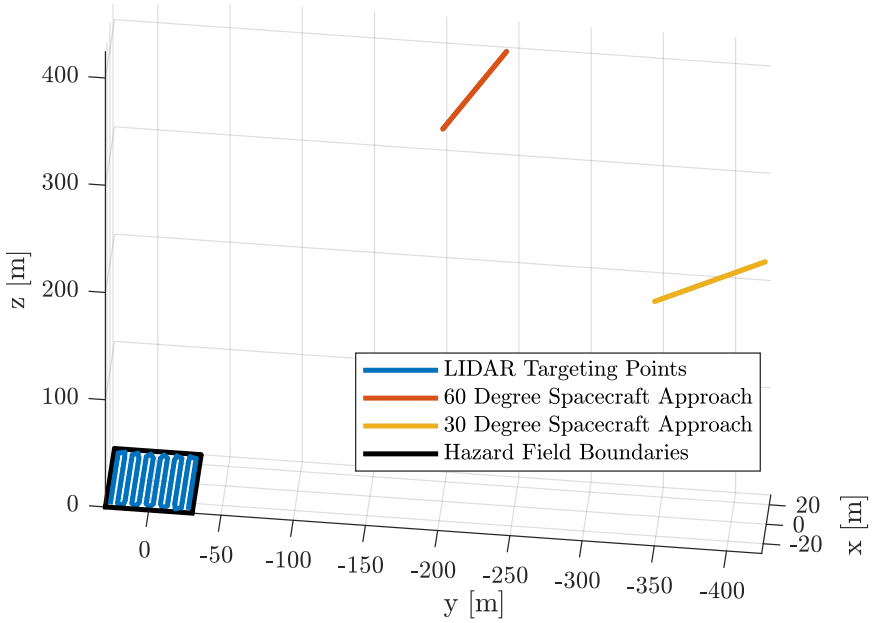
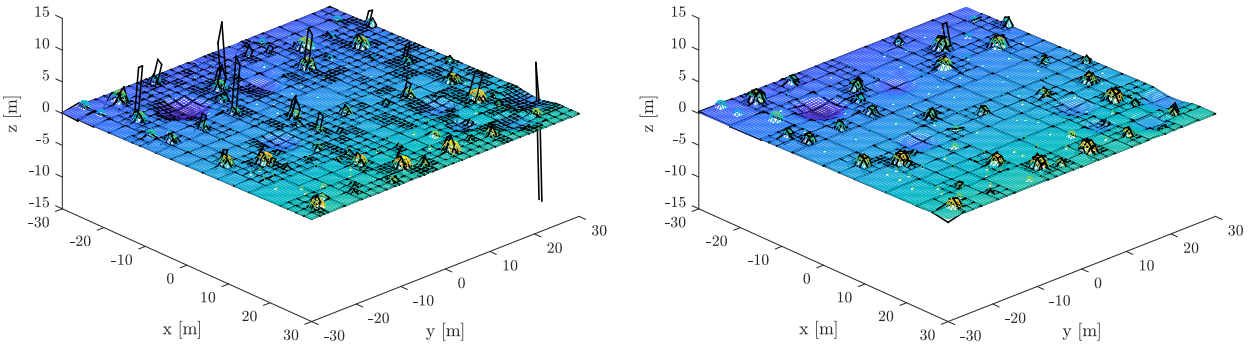


Figure 4: The simulation setup. The spacecraft approaches the hazard field along one of the two trajectories. It scans the hazard field by pointing the boresight of the LIDAR at a set of points which are evenly distributed along the LIDAR targeting line.

The simulated pseudo-measurements are used to refine a QuadTree map and model the hazard field. We overlay the QuadTree map onto the hazard field for each trajectory in Figure 5. In the 30 degree approach, mapping performace appears low at the peaks of most rocks where the planar elements extrapolate the rocks

to appear much larger than reality. This effect is a result of the low angle of approach towards the hazard field. The spacecraft approaches from the negative y direction and terrain on the other side of the rocks is hidden from its view. This effect combined with data association errors from measurement noise leads to poor modeling of the rock face hidden from the spacecraft. These performance issues are greatly reduced in the higher angle of approach case.



(a) 30 degree approach

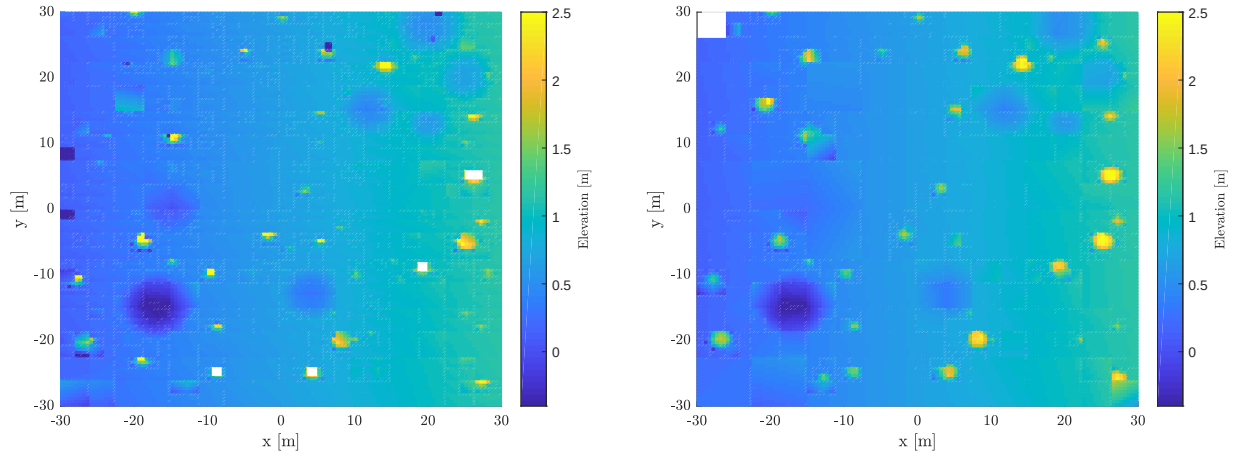
(b) 60 degree approach

Figure 5: The final QuadTree map displayed alongside the truth hazard field. Black lines represent the boundary of each planar element within the QuadTree.

We query the QuadTree to produce the estimated hazard field elevation as shown in Figure 6. To make for an easier comparison with the truth elevation, we saturate the estimates at 2.5 and -0.4 meters. Elevation error is shown in Figure 7. No saturation is performed on the estimate error. We note that all craters and large rocks are represented in the elevation estimates. However, relatively large errors are present in the true elevation of some large rocks in the 30 degree approach. As mentioned before, this is a result of the relatively low angle at which the spacecraft approaches the terrain. Errors are much lower under a 60 degree approach angle as the spacecraft has much better visibility of all rock faces. We also note the absence of small scale rocks from our quadtree. This occurs for one of two reasons; the rocks do not result in a significant enough divergence from planarity to trigger division or the tree has already divided to its minimum resolution in that area.

A key component of our algorithm is surface normal vector estimation. We find the error angle between the true surface normal and our estimated surface normal and display it in Figure 8. In most areas, surface normal error is low. Flat portions of the terrain have errors less than 3 degrees. We do see significant errors in the neighborhood of rocks where the minimum surface resolution is not fine enough to capture every detail of the true surface. We also see relatively higher errors on the rock sides which are obscured from the spacecraft's view in the 30 degree approach case. On flat portions of the map, the 60 degree angle of approach case exhibits lower errors than the 30 degree case.

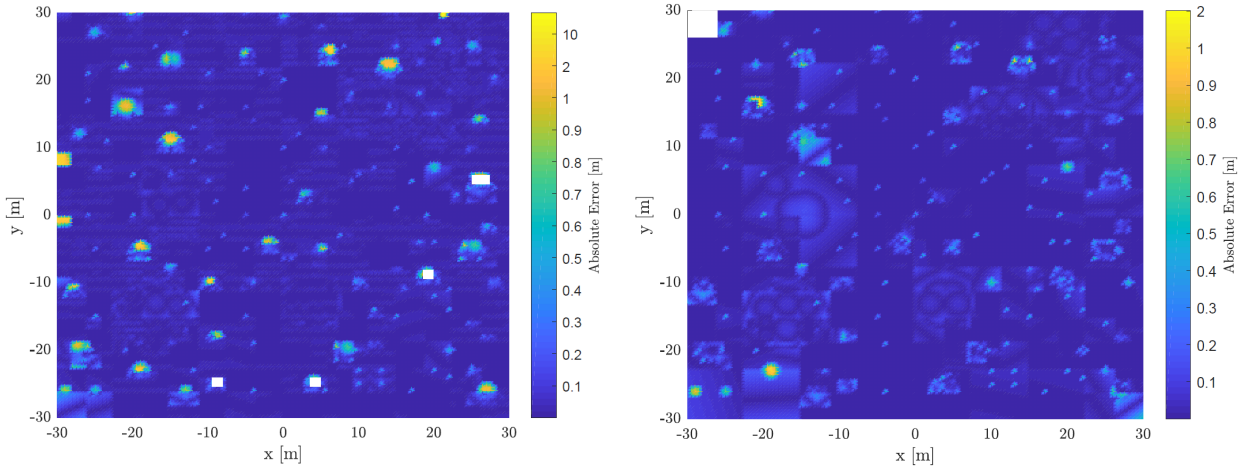
Our system intelligently divides the QuadTree only when the terrain is poorly modeled with a planar estimate. We overlay the divisions of the final QuadTree onto the true hazard field in Figure 9. In general, areas containing rocks or craters exhibit finer cell resolution than areas in which they are absent. The division



(a) 30 degree approach

(b) 60 degree approach

Figure 6: The estimated terrain elevation. White areas do not have a valid planar estimate. Note that the estimates have been saturated at 2.5 and -0.4 meters to ease comparison with the elevation truth.



(a) 30 degree approach

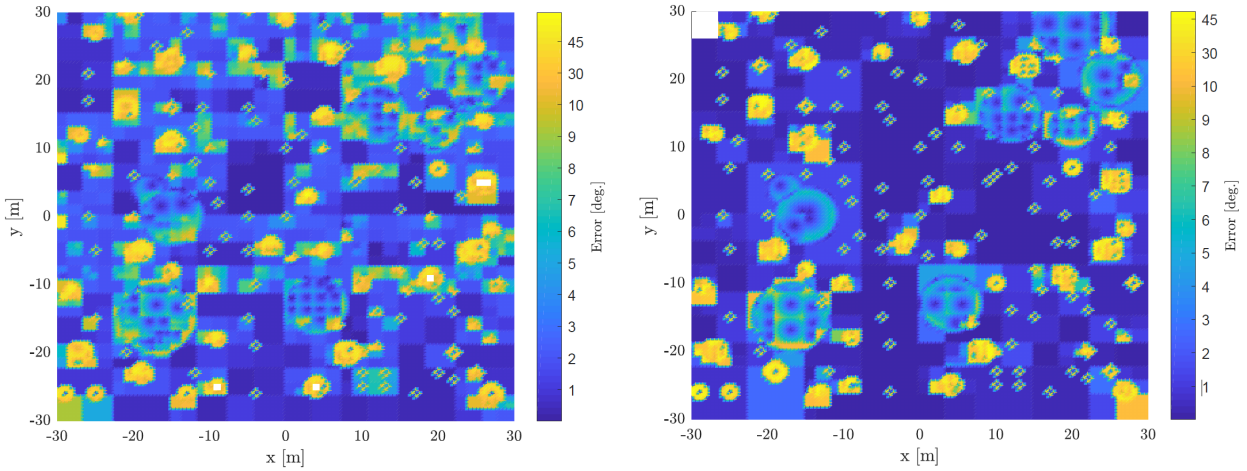
(b) 60 degree approach

Figure 7: The elevation estimate error. White areas do not have a valid planar estimate. Note the color scales of the two approach angles differ.

system is much more accurate in the 60 degree angle of approach case.

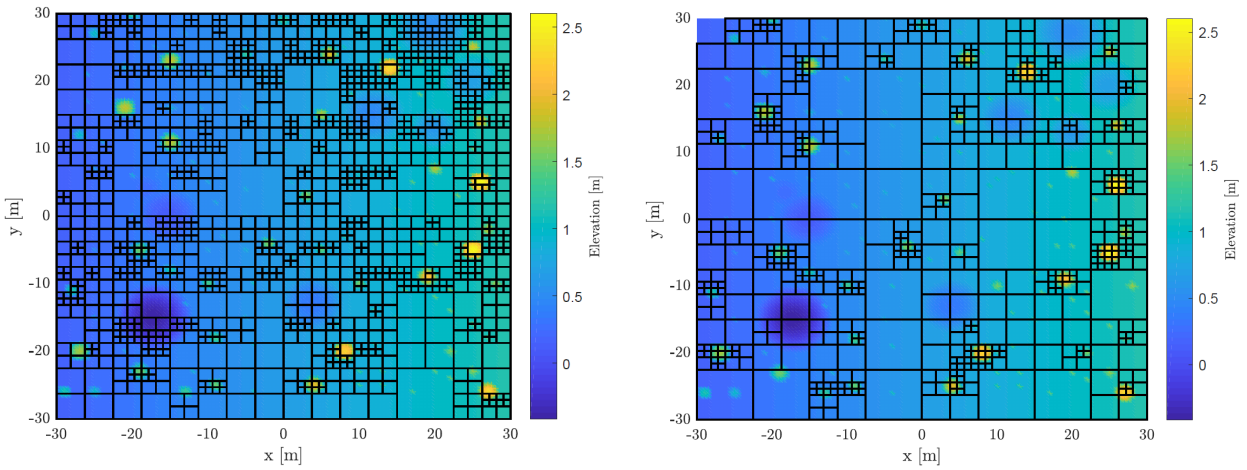
CONCLUSION

We have presented a novel mapping representation for planetary landing scenarios. A target terrain is represented with a variable resolution DEM, or QuadTree, in which each cell contains a planar estimate of its terrain. This representation can be queried with infinite precision and inherently contains estimates of the



(a) 30 degree approach (b) 60 degree approach

Figure 8: The surface normal estimate error. White areas do not have a valid planar estimate.



(a) 30 degree approach (b) 60 degree approach

Figure 9: The elevation truth alongside the QuadTree cell divisions.

terrain's surface normal. These qualities make it uniquely suited for landing site selection algorithms.

Future work will aim to increase the intelligence and adaptability of the mapping algorithm as well as apply it to more realistic simulations or real datasets.

ACKNOWLEDGMENTS

This work was supported by a NASA Space Technology Graduate Research Opportunity, NASA grant number: 80NSSC20K1195.

REFERENCES

[1] U. J. Shankar, W.-J. Shyong, T. B. Criss, and D. Adams, "Lunar terrain surface modeling for the ALHAT program," *2008 IEEE Aerospace Conference*, IEEE, 2008, pp. 1–10.

[2] N. Trawny, A. Huertas, M. E. Luna, C. Y. Villalpando, K. Martin, J. M. Carson, A. E. Johnson, C. Restrepo, and V. E. Roback, "Flight testing a real-time hazard detection system for safe lunar landing on the rocket-powered morpheus vehicle," *AIAA Guidance, Navigation, and Control Conference*, 2015, p. 0326.

[3] T. P. Setterfield, R. A. Hewitt, P.-T. Chen, A. T. Espinoza, N. Trawny, and A. Katake, "LiDAR-Inertial Based Navigation and Mapping for Precision Landing," *2021 IEEE Aerospace Conference (50100)*, 2021, pp. 1–19.

[4] R. Finkel and J. Bentley, "Quad Trees: A Data Structure for Retrieval on Composite Keys.," *Acta Inf.*, Vol. 4, 03 1974, pp. 1–9.

[5] "Autonomous Landing Hazard Avoidance Technology (ALHAT)," `https://techport.nasa.gov/view/94147`. [Online; accessed: October 6th, 2019].

[6] "Safe and Precise Landing – Integrated Capabilities Evolution (SPLICE)," `https://gameon.nasa.gov/projects/safe-and-precise-landing-integrated-capabilities-evolution-splice/`. [Online; accessed: October 13th, 2019].

[7] A. Nelessen, C. Sackier, I. Clark, P. Brugarolas, G. Villar, A. Chen, A. Stehura, R. Otero, E. Stilley, D. Way, K. Edquist, S. Mohan, C. Giovingo, and M. Lefland, "Mars 2020 Entry, Descent, and Landing System Overview," *2019 IEEE Aerospace Conference*, 2019, pp. 1–20.

[8] N. Serrano, "A Bayesian framework for landing site selection during autonomous spacecraft descent," *IEEE International Conference on Intelligent Robots and Systems*, 2006, pp. 5112–5117.

[9] B. E. Cohanim and B. K. Collins, "Landing point designation algorithm for lunar landing," *Journal of Spacecraft and Rockets*, Vol. 46, No. 4, 2009, pp. 858–864.

[10] P. Cui, D. Ge, and A. Gao, "Optimal landing site selection based on safety index during planetary descent," *Acta Astronautica*, Vol. 132, No. October 2016, 2017, pp. 326–336.

[11] M. E. Luna, E. Almeida, G. Spiers, C. Y. Villalpando, A. E. Johnson, and N. Trawny, "Evaluation of the simple safe site selection (S4) hazard detection algorithm using helicopter field test data," *AIAA Guidance, Navigation, and Control Conference, 2017*, No. January, 2017, pp. 1–14.

[12] D. Mango, R. Opromolla, and C. Schmitt, "Hazard detection and landing site selection for planetary exploration using LIDAR," *2020 IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2020 - Proceedings*, 2020, pp. 392–397.

[13] C. Marcus, T. Setterfield, R. Hewitt, and P.-T. Chen, "Landing Site Selection with a Variable-Resolution SLAM-Refined Map," *IEEE Aerospace Conference*, 2022.

[14] J. R. Carpenter and C. N. D'Souza, "Navigation filter best practices," 2018. NASA/TP–2018–219822.

[15] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds," *Information Fusion*, Vol. 14, No. 1, 2013, pp. 57–77.

[16] D. Viejo and M. Cazorla, "3D plane-based egomotion for SLAM on semi-struetured environment," *IEEE International Conference on Intelligent Robots and Systems*, 2007, pp. 2761–2766.

[17] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM*, 2001, pp. 145–152.

[18] A. Zureiki and M. Devy, *SLAM and data fusion from visual landmarks and 3D planes*, Vol. 41. IFAC, 2008.

[19] F. Servant, P. Houlier, and E. Marchand, "Improving monocular plane-based SLAM with inertial measures," *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2010, pp. 3810–3815.

[20] W. S. Grant, R. C. Voorhies, and L. Itti, "Efficient Velodyne SLAM with point and plane features," *Autonomous Robots*, Vol. 43, No. 5, 2019, pp. 1207–1224.

[21] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, Vol. 31, No. 2, 2012, pp. 216–235.

[22] M. J. Gallant and J. A. Marshall, "The LiDAR compass: Extremely lightweight heading estimation with axis maps," *Robotics and Autonomous Systems*, Vol. 82, 2016, pp. 35–45.

[23] M. J. Gallant, *Axis mapping: the estimation of surface orientations and its applications in vehicle localization and structural geology*. PhD thesis, Queen's University (Canada), 2016.

[24] R. A. Hewitt, *Intense navigation: Using active sensor intensity observations to improve localization and mapping*. PhD thesis, Queen's University (Canada), 2018.